Project 4

In this project, you will be writing a program to generate grade reports for a student (or an entire class) from a `csv` file (comma separated values) containing grades. This project will test many of the skills you've acquired this semester, and will introduce you to the `csv` module.

Sample Program Execution

The user should be prompted for a file containing the grades, and given the option to either print a grade report for a specific student or produce a summary grade report for all of the students:

```
Welcome to the CS1064 Grade Report program!
Enter the grades file: grades.csv
-----------------------------------------
Generate a grade report for a single student(R)?
Or summary for all students (S)?: R
-----------------------------------------
Enter student name in the format Last, First: Hokie, Joe
Include only assignments containing (blank for all assignments):
Enter file name for report: JoeHokie.txt
Writing report to JoeHokie.txt...
-----------------------------------------
Go again? (Y/N): Y
-----------------------------------------
Generate a grade report for a single student(R)?
Or summary for all students (S)?: S
-----------------------------------------
Include only assignments containing (blank for all assignments): HW
Enter file name for report: Grades.txt
-----------------------------------------
Go again? (Y/N): N
```

The line:

```
Include only assignments containing (blank for all assignments):
```

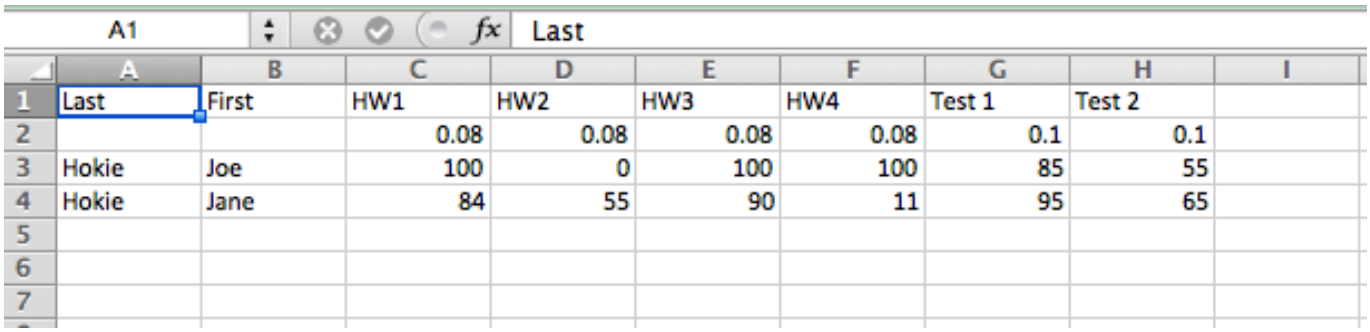
Lets the user filter out unwanted assignments. In the above example, whoever is running the program wants all of the assignments included (since they typed whitespace, or just pressed "enter") for student Joe Hokie, but ONLY wants assignments whose names contain `"HW"` included in the summary report. In this case (when `"HW"` is entered) all of the other assignments would be excluded from the grade calculation.

The assignment names in the `csv` file can be anything and the user can enter any string (multiple words, etc.), so you'll need to read in the assignment names from the file and check to see if they contain the user entered string. In this example `"HW1"`, `"HW2"`, and `"Late HW"` would all be counted when calculating grades, but an assignment called `"Homework 1"` would not.

Input File Format
The file in the sample execution above (`grades.csv`) is in `csv` format, a common file format for distributing data. If you open the file `grades.csv` in a text editor it will look something like this:

```
Last,First,HW1,HW2,HW3,HW4,Test 1,Test 2
,,0.08,0.08,0.08,0.08,0.1,0.1
Hokie,Joe,100,0,100,100,85,55
Hokie,Jane,84,55,90,11,95,65
```

So a text editor will show you all of the data, but it's not the most readable way to look at the file. A better way to visualize what's in the file is to open it in excel:



	A	B	C	D	E	F	G	H	I
1	Last	First	HW1	HW2	HW3	HW4	Test 1	Test 2	
2			0.08	0.08	0.08	0.08	0.1	0.1	
3	Hokie	Joe	100	0	100	100	85	55	
4	Hokie	Jane	84	55	90	11	95	65	
5									
6									
7									

The first 2 columns will always be last name and first name respectively, after that there may be an arbitrary number of columns containing an assignment name, a weight (between 0 and 1), and then a grade for each student. The weights "row" will be blank in the last and first name columns. Further, there can be any number of students in the file.

Output File Formats
For a single student's grade report your program should output all of the **selected** students assignments, their overall grade for those assignments, and the class average for those assignments. So for the above execution `JoeHokie.txt` would contain grades for all of the assignments:

```
Hokie, Joe
---------------------------------
Assignment       Weight         Grade
HW1              8.0%           100
HW2              8.0%           0
HW3              8.0%           100
HW4              8.0%           100
Test 1           10.0%          85
Test 2           10.0%          55
---------------------------------
Overall:         38.0%/52.0%    73.1
Class Average:   36.6%/52.0%    70.4
```

While the summary report would only include grades with "HW" the assignment name. So Grades.txt would contain:

```
Summary Report
---------------------------------
Hokie, Joe     24.0%/32.0%    75.0
Hokie, Jane    19.2%/32.0%    60.0
---------------------------------
Class Average: 21.6%/32.0%    67.5
```

In both of these examples, the grades are calculated using a weighted average. When all of the assignments are included the following calculation is performed for Joe Hokie:

```
38.0 = 0.08*100 + 0.08*0 + 0.08*100 + 0.08*100 + 0.10*85 + 0.10*55
```

The total of all of the weights is 52.0, so Joe Hokie's overall average is 38.0/52.0 = 73.1. The same process is used for just the HW grades:

```
24.0 = 0.08*100 + 0.08*0 + 0.08*100 + 0.08*100
```

Out of 32.0 total points for HW category. In this case, Jane Hokie's grade (and any other students in the file) would also be computed.

The formats for these two types of files won't change when different assignments are selected, only what's included in the calculation changes.

Hints

For this project I'd recommend reading the file in a different way. Rather than reading the file manually, use the csv module:

```
import csv
f = open("grades.csv", "r")
r = csv.reader(f)

for row in r:
     print(row)
```

Each row is a list containing the data from one row in the csv file. This code would print:

```
['Last', 'First', 'HW1', 'HW2', 'HW3', 'HW4', 'Test 1', 'Test 2']
['', '', '0.08', '0.08', '0.08', '0.08', '0.1', '0.1']
['Hokie', 'Joe', '100', '0', '100', '100', '85', '55']
['Hokie', 'Jane', '84', '55', '90', '11', '95', '65']
```

Another useful function is `next().next()` allows us to pull individual rows out of the `csv` file without using a for loop. The ordering is the same as the for loop though:

```
import csv
f = open("grades.csv", "r")
r = csv.reader(f)

# Get just the first row:
# ['Last', 'First', 'HW1', 'HW2', 'HW3', 'HW4', 'Test 1', 'Test 2']
assignments = next(r, [])

# Get just the second row:
# ['', '', '0.08', '0.08', '0.08', '0.08', '0.1', '0.1']
weights = next(r, [])

# for loop for the rest of the data
for row in r:
     print(row)
```

The for loop in this case would just print the "data" rows:

```
['Hokie', 'Joe', '100', '0', '100', '100', '85', '55']
['Hokie', 'Jane', '84', '55', '90', '11', '95', '65']
```

You may use any function that's part of the `csv` package when working on this assignment. For more information check out the python documentation:

https://docs.python.org/3.4/library/csv.html

What to Submit

For this assignment you should submit your **p4.py** file. Your file must be named **p4.py**.

This assignment will be graded automatically. Test your programs thoroughly before submitting them. Make sure that your programs produce correct results for every logically valid test case you can think of. Do not waste submissions on untested code, or on code that does not compile with the supplied code from the course website.

Web-CAT will assign a score based on runtime testing of your submission; your best score will be counted; the TAs will later verify that your best submission meets the stated restrictions, and assess penalties if not.

To submit this assignment:

1. Visit http://web-cat.cs.vt.edu in your web browser.
2. Enter your Virginia Tech PID and password in the appropriate fields on the log-in screen, and make sure that **Virginia Tech** is selected as the institution. Click **Login**.

3. The Web-CAT home screen will display useful announcements and assignments that are currently accepting submissions. Find the assignment that you want to submit in the table, and click the "Submit" button next to it.
4. Click the **Browse...** button and select the file you want to upload. The homework assignments and programming projects for this course should be self-contained in a single **.py** file, so you can simply select that one file.
5. Click the **Upload Submission** button. The next page will ask you to review your selection to ensure that you have chosen the right file. If everything looks correct, click **Confirm**.

The next page will show that your assignment is currently queued for grading, with an estimated wait time. This page will refresh itself automatically, and when grading is complete you will be taken to a page with your results.

When your results are ready, make sure that you have **80%** on the assignment. If you have anything less, read the hints that Web-CAT gave you and make any corrections to your code that you need to make, then submit again. Remember that for the programming projects in this class (as opposed to the homework assignments), you can submit up to 5 days after the due date, with a 10% penalty per day late.

Pledge

Each of your program submissions must be pledged to conform to the Honor Code requirements for this course. Specifically, you **must** include the following pledge statement in the submitted file:

```
#  <include a description of the purpose of this file/project/package>
#
#  @author <name and surname> (your VT PID)
#  @date   <the date>
#
# Virginia Tech Honor Code Pledge
#  On my honor:
#
#  - I have not discussed the Python language code in my program with
#    anyone other than my instructor or the teaching assistants
#    assigned to this course.
#  - I have not used Python language code obtained from another student,
#    or any other unauthorized source, either modified or unmodified.
#  - If any Python language code or documentation used in my program
#    was obtained from another source, such as a text book of coarse
#    notes, that has been clearly noted with a proper citation in
#    the comments of my program.
#  - I have not designed this program in such a way as to defeat or
#    interfere with the normal operation of the Web-Cat Server.
#
#  <your name>
```

Failure to include this pledge in a submission will result in the submission being disallowed during code review.